

Extracting Keywords with Combination of Text Categorization by Graph based KNN

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the graph based KNN version as the approach to the text classification and the keyword extraction, for automating fully the keyword extraction. In the previous work, we proposed the KNN version as an approach to the keyword extraction, but a domain should be given as a tag, in addition to an input text. In this research, we adopt the graph based KNN version which takes a graph, instead of a numerical vector, as the approach to both tasks. In the proposed system, only text without its domain tag is given as the input, its domain is assigned to the text through the text classification, and its keywords are extracted by the classifier which correspond to the domain. The goal of this research is to automate the keyword extraction without presenting a domain tag and improve the performance of both tasks, using the graph based KNN version.

I. INTRODUCTION

The keyword extraction is the process of extracting automatically important words as representatives of a given text. It is interpreted into a binary classification where each word is classified into keyword or non-keyword. The process of keyword extraction is to index a text which is given the input into a list of words, classify each word into one of the two categories, and extract words which are classified into keyword as the output. The classification task which is mapped from the keyword extraction is a domain dependent task where a same entity may be classified differently, depending on the domain. The domain should be manually presented as a text for executing the keyword extraction.

This research is motivated by the successful results from applying the graph based KNN algorithm to both the keyword extraction and the text classification. The keyword extraction is a domain dependent classification where a same entity may be classified differently depending on the domain, whereas the text classification is a domain independent classification where a same entity is always classified identically independent of the domain. It is required to know the domain to the input text in advance for executing the keyword extraction. The process of assigning a domain to a text is automated through the text classification. We connect the keyword extraction with the text classification for solving the problem.

The graph based KNN algorithm is adopted for implementing both the text categorization and the keyword extraction. The similarity metric between graphs is defined, and the KNN algorithm is modified into the graph-based version, based on the similarity metric. The modified KNN algorithm is applied to the text classification which classifies an input text into its own domain. In the process of keyword extraction, an input text is indexed into words, each word is classified into keyword or non-keyword, and ones which are classified with keyword are extracted. In this research, the process of deciding the domain to the input text from which keywords are extracted is automated by the text categorization.

Let us mention some benefits from this research. The problems in encoding words or texts into numerical vectors are avoided by encoding them into graphs. The performance in both tasks, the text classification and the keyword extraction, is improved by adopting the graph based KNN algorithm. We are free from presenting the domain as a tag by connecting the keyword extraction with the text classification. Automating the process of nominating a classifier which corresponds to a domain is to upgrade the keyword extraction system.

Let us mention the remaining tasks for upgrading this research. More operations on graphs may be defined and characterized mathematically. Texts and words may be encoded into compound representations, each of which consists of more than one structured form. Other machine learning algorithms and deep learning algorithms may be modified into graph-based versions. The keyword extraction system may be implemented as a real program or a library module by adopting the graph based KNN algorithm.

II. PROPOSED APPROACH

A. Similarity Metric

This section is concerned with the graph as a word representation and the similarity between graphs. In representing a word into a graph, a vertex indicates a text, and an edge indicates a similarity between texts. The similarity between edges is computed, before computing the similarity between graphs, viewing a graph as a set of edges. The

similarity between graphs is computed by averaging over the similarities of all possible pairs of edges. In this section, we describe the process of encoding a word into a graph and the process of computing the similarity between graphs.

The process of encoding words into graphs is illustrated in Figure 1. In the vertex definition, from a text collection, texts which are relevant to the word are retrieved as the vertices. In the edge definition, the similarities among the retrieved texts are computed as edges. In the edge selection, the edges with their higher similarities are selected for representing a graph. Refer to [1] for studying the encoding process in more detail.

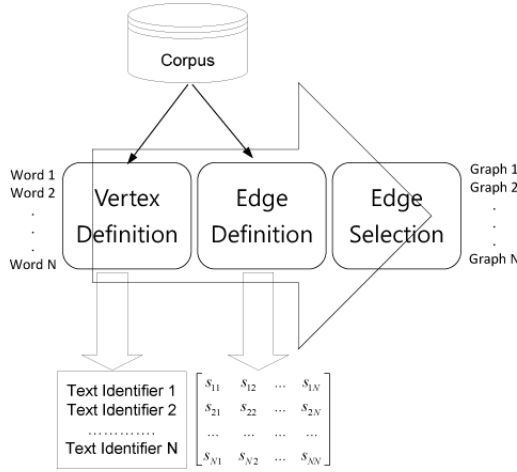


Figure 1. Process of Encoding Words into Graphs

The three cases of computing the similarity between two edges are illustrated in Figure 2. Each weight which is associated with its own edge is assumed to be a normalized value between zero and one, and if both nodes are identical to each other, the average over two weights is the similarity between two edges. If either of the two nodes is identical to each other, the product of two weights is the similarity between two edges. If neither of two nodes is identical, zero is the similarity between them. The similarity between two edges is always a normalized value between zero and one.

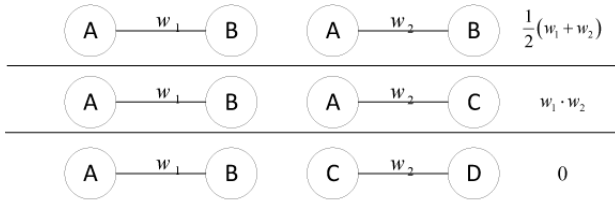


Figure 2. Three Cases of Comparing Two Edges with Each Other

Let us mention the process of computing the similarity between graphs as a semantic similarity between words. The two graphs which represent words are notated by edge sets, $G_1 = \{e_{11}, e_{12}, \dots, e_{1|G_1|}\}$ and $G_2 =$

$\{e_{21}, e_{22}, \dots, e_{2|G_2|}\}$. Two edges, $e_{1i} \in G_1$ and $e_{2j} \in G_2$, are associated with their own weights, w_{1i} and w_{2j} , and the similarity between two edges, e_{1i} and e_{2j} by equation (1), (2), or (3), depending on the cases which are mentioned above,

$$\text{sim}(e_{1i}, e_{2j}) = \frac{1}{2}(w_{1i} + w_{2j}) \quad (1)$$

$$\text{sim}(e_{1i}, e_{2j}) = w_{1i} \times w_{2j} \quad (2)$$

$$\text{sim}(e_{1i}, e_{2j}) = 0 \quad (3)$$

The similarity between two graphs is computed by equation (4),

$$\text{sim}(G_1, G_2) = \frac{1}{|G_1| \times |G_2|} \sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} \text{sim}(e_{1i}, e_{2j}). \quad (4)$$

The value which is generated from equation (4), is always given as a normalized value between zero and one as shown in equation (5),

$$0 \leq \text{sim}(G_1, G_2) \leq 1. \quad (5)$$

Let us make some remarks on the process of encoding a word into a graph and computing the similarity between graphs. A word is encoded into a graph by the vertex definition, the edge definition, and the edge weighting. The three cases are considered for computing the similarity between edges. The similarity between graphs is computed by equation (4). In the future research, we consider representing a word into multiple graphs.

B. Graph based KNN Algorithm

This section is concerned with the graph based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [1]. The similarity metric is used for computing the similarity between a novice graph and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into graphs, and they are notated by a set, $Tr = \{G_1, G_2, \dots, G_N\}$. A novice word is encoded into a graph notated by G . Its similarities with the graphs which represent the sample words are computed by equation (4), as $\text{sim}(G, G_1), \text{sim}(G, G_2), \dots, \text{sim}(G, G_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after

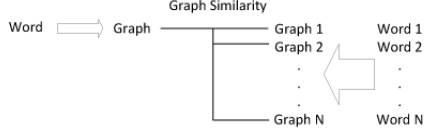


Figure 3. Similarities of Novice Input with Training Examples

computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (6),

$$Nr = \text{Select}_k(Tr, G), Nr \subseteq Tr \quad (6)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (7),

$$Nr = \{G_1^{near}, G_2^{near}, \dots, G_k^{near}\} \quad (7)$$

The elements in the set, Nr , are used for voting their labels in the next step.

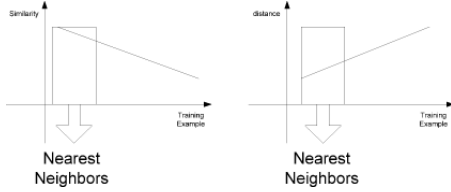


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(G_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, G , is decided by equation (8),

$$\text{label}(G) = \arg\max_{j=1}^m \text{Count}(Nr, c_j) \quad (8)$$

The process of deciding the label of a novice item by equation (8), is called voting.

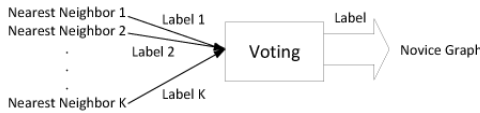


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the graph-based version of KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its

nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the keyword extraction system which adopts the graph based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the keyword extraction. It is viewed as a binary classification which classifies a word into keyword or non-keyword. This section is intended to describe the keyword extraction system with respect to its function and its architecture.

The process of collecting the sample words and encoding them into graphs for implementing the keyword extraction system is illustrated in Figure 6. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

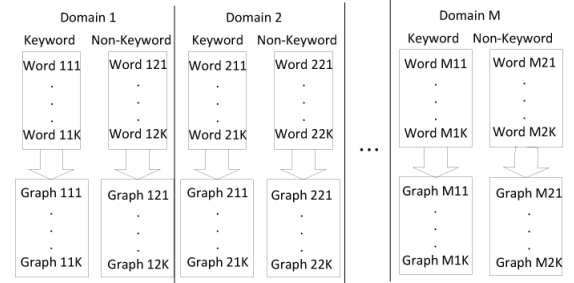


Figure 6. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, sample words in the keyword group and the non-keyword group and ones which are indexed from the text are mapped into graphs. The words which are indexed from the text are classified into one of the two categories in the similarity computation module and the voting module. The system consists of the four modules: indexing module, encoding module, similarity computation module, and voting module.

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which

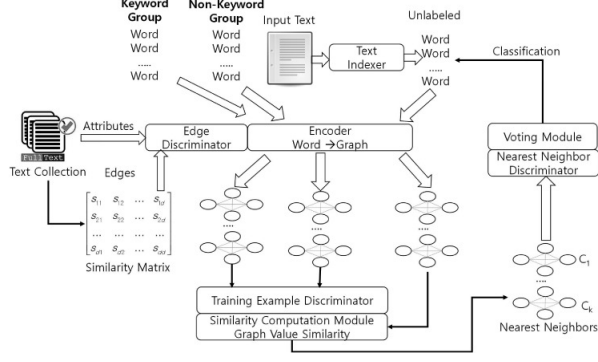


Figure 7. System Architecture

are labeled with keyword or non-keyword are collected within a domain, and they are encoded into graphs. An input text is indexed into a list of words, and they are also encoded into graphs. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. The words which are classified into keyword are extracted as the final output.

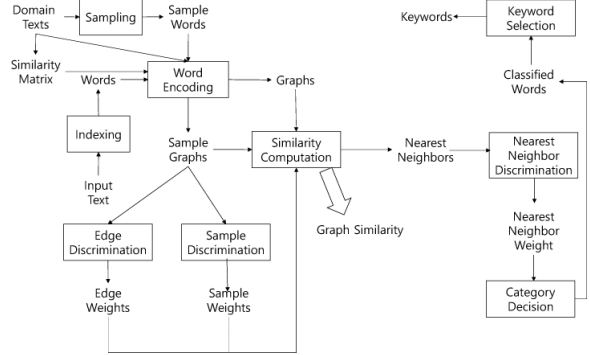


Figure 8. System Flow

Let us make some remarks on the proposed system whose architecture is presented in Figure 7. The keyword extraction is defined as a binary classification where each word is classified into keyword or non-keyword. Each word is encoded into a graph instead of a numerical vector and is classified directly. Among words which are included in the input text, ones which are classified into keyword are selected. We need to add the text classification module for predicting the domain of the input text automatically.

D. Connection with Text Classification

This section is concerned with the connection of the keyword extraction with the text classification. In the previous section, we mentioned the keyword extraction system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN

algorithm which was described in Section II-B is used for implementing the text classification. This section is intended to describe the connection of the keyword extraction system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into graphs by the process which is described in Section II-A. The similarity computation module is for computing the similarities between graphs which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

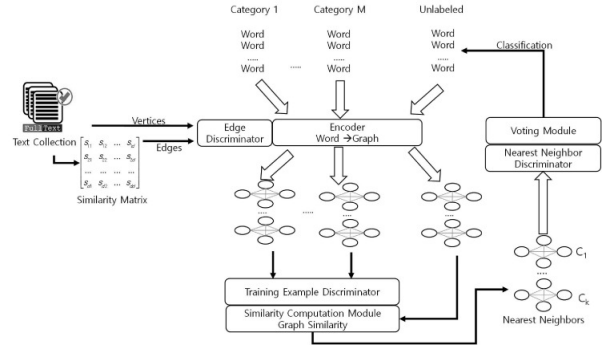


Figure 9. Text Categorization System

The connection of the keyword extraction with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the keyword extraction system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the keyword extraction, automatically.

The process of executing the keyword extraction, connecting with the text classification. Sample texts each of which is labeled with its own domain are prepared, and sample words each of which is labeled with keyword or non-keyword are prepared in each domain. A novice text is classified into one among the predefined domains, and the classifier which corresponds to the domain is nominated. The novice text is indexed into a list of words, and each word is classified into keyword or non-keyword. The list of words which is classified into keyword is the final output from this system; the domain which is assigned to the input text is the guide for selecting a classifier.

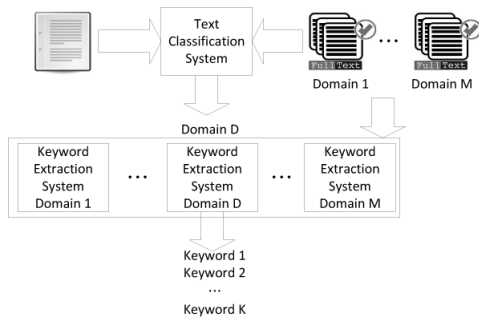


Figure 10. Keyword Extraction System connected with Text categorization

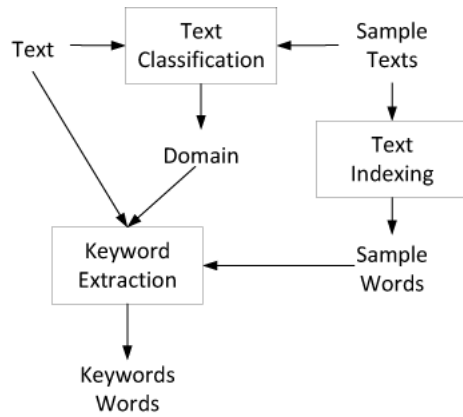


Figure 11. System Flow of Keyword Extraction connected with Text Categorization

Let us make some remarks on the connection of the keyword extraction with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the keyword extraction is to extract important words from a text. In the execution flow, the input text is classified into a domain, it is indexed into a list of words, and each word is classified into keyword or non-keyword. We consider connecting the text classification as the main task the keyword extraction as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Comparing Graph based K Nearest Neighbor with Traditional Version in Word Categorization in NewsPage.com, 12-18, International Journal of Advanced Social Sciences, 1, 2018.
- [2] T. Jo, "Graph based K Nearest Neighbors for Keyword Extraction in Current Affair Domain", 47-48, The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence, 2018.